令和4年度(令和4年4月入学)
博士前期課程(修士課程)一般入試(第Ⅰ期)
令和3年度(令和3年秋入学)
博士前期課程(修士課程)外国人留学生特別入試
情報工学専攻

数　　学　(120分)

〔注意事項〕

1. 監督者の指示があるまで，問題冊子(この冊子)を開いてはいけません。
2. 配布物は，この問題冊子1部，解答用紙3枚と計算用紙1枚です。
3. 解答用紙には志望専攻名，受験番号を記入する欄がそれぞれ1箇所ずつあります。監督者の指示に従って，すべての解答用紙(合計3枚)の志望専攻名欄と受験番号欄に志望専攻名と受験番号を記入しなさい。
4. 解答は，問題番号に対応する解答用紙の指定された場所に書きなさい。解答を解答用紙の裏面に書いてはいけません。解答用紙，計算用紙の追加，交換はしません。
5. 問題は全部で3問あり，2ページにわたって印刷されています。落丁・乱丁および印刷の不鮮明な箇所などがあれば，手をあげて監督者に知らせなさい。
6. 問題冊子の白紙と余白は，計算などに使用してもよろしい。
7. 解答用紙は，持ち帰ってはいけません。
8. 問題冊子と計算用紙は，持ち帰りなさい。

1 a, b を実数とし，4 次正方行列 A を

$$A = \begin{pmatrix} 0 & 2 & -3 & 1 \\ 1 & 1 & b & 0 \\ -2 & b & -10 & 2 \\ 1 & a & 5 & a \end{pmatrix}$$

により定める。A の階数は 2 であるとする。このとき，a, b の値を求めよ。また，$\mathbf{R}^4$ の一次変換 $T_A(\boldsymbol{x}) = A\boldsymbol{x}$ $(\boldsymbol{x} \in \boldsymbol{R}^4)$ の固有値 0 に属する固有ベクトルをすべて求めよ。

2 関数 $f(x) = \dfrac{1 - e^{-2x}}{x\sqrt{x}}$ $(x > 0)$ について以下の問に答えよ。

(1) 極限 $\lim_{x \to +0} f(x)$ を求めよ。

(2) 実数 x に対し，不等式 $1 - e^{-2x} \leqq 2x$ を示せ。

(3) 広義積分 $\int_0^1 f(x)\,dx$ が収束すること，および $\int_0^1 f(x)\,dx \leqq 4$ であることを示せ。

(4) 広義積分 $\int_1^\infty f(x)\,dx$ が収束すること，および $\int_1^\infty f(x)\,dx \leqq 2$ であることを示せ。

3 (1) k を 2 以上 12 以下の自然数とする。サイコロを二回投げるとき，一回目に出る目と二回目に出る目の和が k であるという事象を A_k とし，二回目に出る目が 1 であるという事象を B とする。このとき，A_k と B が独立であるような k をすべて求めよ。

(2) c を実数とする。2 次元確率変数 (X, Y) の同時確率密度関数は

$$f(x,y) = \begin{cases} ce^{-\frac{1}{2}(x+y-2)} & (x \geqq 1 \text{ かつ } y \geqq 1 \text{ のとき}) \\ 0 & (x < 1 \text{ または } y < 1 \text{ のとき}) \end{cases}$$

であるとする。このとき，c の値および，X の期待値 $E(X)$ を求めよ。

(3) ある機種のスマートフォンの重さ X（グラム）は，母平均 μ，母分散 0.36 の正規分布 $N(\mu, 0.36)$ に従う。その機種のスマートフォン 36 台の重さを測定したところ，測定値の平均は 161.2010 であった。このとき，μ の信頼度 95 %の信頼区間を求めよ。ただし，正規分布 $N(\mu, \sigma^2)$ に従う確率変数 Y に対し，$\dfrac{Y-\mu}{\sigma}$ は $N(0,1)$ に従うことを用いてよく，また，$N(0,1)$ の上側 α 点を $z(\alpha)$ と表すとき，近似値として

$$z(0.05) = 1.645,\ z(0.025) = 1.960,\ z(0.01) = 2.326,\ z(0.005) = 2.576$$

を用いてよい。

（以上）

令和４年４月入学 一般入試Ⅰ期 April Admission for the 2022 Academic Year for Japanese Students (1st period)
令和３年秋入学 外国人留学生特別入試 Fall Admission for the 2021 Academic Year for International Students
京都工芸繊維大学 大学院工芸科学研究科 博士前期課程（修士課程） 情報工学専攻 試験問題
Question booklet of Entrance Examination for Master's Program of Information Science,
Graduate School of Science and Technology, Kyoto Institute of Technology (KIT)

専門科目 Special Subject

[注意事項 Cautions]

1. この問題冊子は合図があるまで中を開かないでください。この中身は以下の２題であり，２題とも必須です。落丁・乱丁および印刷の不鮮明な箇所などがあれば、手を挙げて監督者に知らせなさい。
   Do not open this question booklet until permitted by the proctor. Answer all two subject parts listed below. Raise your hand and inform the proctors of any missing pages, disarranged pages, unclear printing, etc.
   プログラミング（１） Programming (1)
   プログラミング（２） Programming (2)
2. 配布物は、この問題冊子１部、解答用紙２枚、および下書き用紙１枚です。
   The proctors distribute this question booklet, two answer sheets, and one memo sheet.
3. 机の上には受験票以外に、次のものを置いてもよろしい。
   You may put the following goods in addition to your exam admission ticket.
   (1) 黒鉛筆とシャープペンシル Black pencils and mechanical pencils
   (2) プラスチック製の消しゴム Plastic erasers
   (3) 電動でない小型の鉛筆削り Small-sized non-electric pencil sharpeners
   (4) 秒針音がしない小型の時計（辞書、電卓、通信等の機能があるものは不可）
   Small-sized silent watches or clocks without any additional functions such as dictionary, calculator, communication, etc.
   (5) 眼鏡、ハンカチ、目薬、ティッシュペーパー（袋又は箱から中身だけを取り出したもの）
   Glasses, handkerchiefs, eye drops, tissues without package
   これら以外については監督者の了解を受けてください。
   Ask the proctors for permission to use any goods other than the above.
4. プログラミング（１）とプログラミング（２）を別の解答用紙に解答してください。解答用紙２枚すべての上欄指定枠内に、問題科目名と問題番号、志望専攻名、受験番号を忘れずに記入してください。解答用紙の裏面に解答を書いても構いません。解答用紙と下書き用紙の追加配布はしません。
   Use a separate answer sheet for each subject part. Fill in the subject-part name, the major of Master's Program, and your examinee's number in the designated boxes on all two answer sheets. You can use both sides of the answer sheet. No additional sheet is available.
5. この問題冊子はバラしても構いません。 You can unbind this booklet.
6. 試験終了後も退出の許可があるまで退室はできません。中途退室できません。
   Do not leave the room after the exam until permitted by the proctor. Also, you do not during the exam.
7. 問題冊子と下書き用紙は持ち帰ってください。
   Bring this question booklet and the memo sheets when you leave the room after the exam.

プログラミング(1)　[1/2]

問 1　C 言語で記述された Program1 を実行したとき、①、②、③の行にある printf 文による出力(output)をそれぞれ示せ。

Program1

```
int main(void){
  char *city[] = {"Kyoto", "Osaka", "Kobe"}, **p = &city[2];
/* ① */ printf("%c¥n", city[2][3] == 'a' ? 'Y' : 'N');
/* ② */ printf("%c¥n", **p+1);
/* ③ */ printf("%c¥n", (*--p)[3]);
  return 0;
}
```

問 2　C 言語で記述された Program2 は、整数(integer)の配列(array) arr における要素(element)の中で、n 番目に大きい値を返す関数(function) nthmax である。len は配列 arr の要素の数、

int argmax(int ray[], int last)

は、ray[0]から ray[last]までの中で最大となる要素の添字(subscript)を返す関数である。ただし、n は 1 以上 len 以下の整数とする。下記の(あ)、(い)を埋めて関数 nthmax を作成せよ。

Program2

```
int nthmax(int arr[], int len, int n){
  int arg, i;
  for(i = 1; i < n; i++){
    arg = argmax( (あ) );
    (い) = arr[len-i];
  }
  return arr[argmax( (あ) )];
}
```

問 3　C 言語で記述された Program3 は、文字列(string) str1 の後に文字列 str2 を n 文字だけ付け加える関数 strcatn である。ただし、n は 0 以上 str2 の文字数以下の整数とする。下記の(う)、(え)、(お)を埋めて、この関数を作成せよ。

Program3

```
void strcatn(char *str1, const char *str2, int n){
  for(; *str1 != '¥0'; (う) ){ }
  while( (え) ){
    (お) = *str2++;
  }
  *str1 = '¥0';
}
```

[次ページに続く]

プログラミング(1)　[2/2]

問 4　Java 言語で記述された Program4 に関する(a)～(c)の問いに答えよ。

Program4

```
class Abst{
  int x;
  Abst(){ x = 0; }
  void pre(int dv){}
  void change_x(int dx){}
  public void update(int dv, int dx){
    pre(dv);
    change_x(dx);
    System.out.println(x);
  }
  public void update(int dx){ update(10, dx); }
}

class MetA extends Abst{
  MetA(){ super(); }
  void change_x(int dx){ x += dx; }
}

class MetB extends MetA{
  private int v = 1;
  MetB(){ super(); }
  void pre(int dv){ v += dv; }
  void change_x(int dx){ super.change_x(v + dx); }
}

class Main{
  public static void main(String args[]){
    Abst met = new MetA();
    met.update(5);
    met = new MetB();
    met.update(1, 2);
    met.update(3);
  }
}
```

(a) クラス(class) Abst のメソッド(method) Abst のように、インスタンス(instance)を生成する際に呼び出されて、インスタンス変数(variable)の初期化などを行うメソッドを何と呼ぶか。

(b) クラス Abst のメソッド update のように、名前が同じメソッドを複数定義することを何と呼ぶか。

(c) Program4 を実行したときの出力を示せ。

プログラミング(2)　[1/3]

問1　二分探索木(binary search tree)に節点(node)を挿入(insert)するC言語プログラムであるProgram1について、以下の問(a)～(d)に答えよ。なお、プログラム中の左端の数字は行番号である。節点は Node 構造体(structure)によって表現される。また、プログラム 28 行目で宣言(declare)されている変数(variable) root は、二分探索木の根(root)へのポインタ(pointer)である。

Program 1

```
 1  #include <stdio.h>
 2  #include <stdlib.h>
 3
 4  typedef struct node {
 5    int value;
 6    struct node *left, *right;
 7  } Node;
 8
 9  void insert(Node **root, int value) {
10    Node **p;
11    p = root;
12    while( ① != NULL) {
13      if( ② )->value == value)
14        return;
15      else if( ② )->value > value)
16        p = ③ )->left;
17      else
18        p = ③ )->right;
19    }
20    if((*p = (Node *)malloc(sizeof(Node))) == NULL)
21      exit(1);
22    else {
23      (*p)->left = NULL; (*p)->right = NULL; (*p)->value = value;
24    }
25  }
26
27  int main(void) {
28    Node *root;
29    root = NULL;
30    insert(&root, 3); insert(&root, 1);
31    insert(&root, 4); insert(&root, 2);
32    printf("%d¥n", root->left->right->value);
33    return 0;
34  }
```

(a)　Program1 の空欄 ① ～ ③ を埋めて、プログラムを完成せよ。

(b)　Program1 が正常に実行されたときの、標準出力(standard output)への出力結果を示せ。

(c)　節点の数が N 個の二分探索木に 1 個の節点を挿入する操作の時間計算量(time complexity)のオーダ(order)を、平均的な場合(average case)と最悪の場合(worst case)それぞれについて示せ。

(d)　配列(array)を対象とした二分探索(binary search)と比べて、二分探索木を用いることの長所(merit)と短所(demerit)をそれぞれ 1 つずつ簡潔に説明せよ。

[次ページに続く]

プログラミング(2)　[2/3]

問2 Boyer-Moore 法（以下、「BM 法」と記す）を用いた文字列検索(string search)に関する問(a)～(d)に答えよ。文字列検索とは、検索対象となる文字列(string)である「テキスト(text)」の中から、特定の文字列「パターン(pattern)」を検索することである。Program2 は、BM 法による文字列検索を行う C 言語プログラムである。プログラム中の左端の数字は行番号である。なお、プログラム中で使用している関数(function) `strlen` は、引数(argument)に指定した文字列の文字数を返す関数である。

これ以降では、文字列検索の例(example)として、テキストが “ABCDABCBAB”、パターンが “CBAB” であるときの、BM 法による検索過程を示す。

(1) BM 法では、テキストとパターンを重ね合わせ、パターンの末尾(tail)から先頭(head)に向かって、順番(sequential)に文字(character)の比較(compare)を行う。そこで、はじめに、図 1 (i) に示すように、比較を行う文字の位置(position) （以下、「比較位置」と呼ぶ）を、パターンの末尾に設定する。なお、図中の矢印(arrow)は比較位置を示している。

(2) テキストの比較位置の文字は “D” であり、パターンには含まれない文字である。この場合は、図 1 (ii) に示すように、パターンの文字数である 4 文字分だけ、比較位置を右に移動(shift)する。

(3) 比較位置にある文字は、テキストとパターンのどちらも “B” である。このように、文字が一致している場合は、図 1 (iii) に示すように、比較位置を 1 文字分左に移動し、次の文字を比較する。

(4) 比較位置にある文字は、テキストでは “C”、パターンでは “A” であり、一致していないが、“C” はパターンに含まれている文字である。テキストの比較位置の文字がパターンに含まれている場合は、その文字がパターンの末尾から数えて何文字目であるかを求める（最初の文字を 0 文字目とする）。“C” は、パターンの末尾から数えて 3 文字目にあるので、図 1 (iv) に示すように、比較位置を 3 文字分右に移動する。

(5) 比較位置から順番にテキストとパターンの文字を比較していくと、パターンのすべての文字がテキスト内の文字と一致している。これにより、パターンがテキストの先頭から 6 文字目以降にあることがわかるため、検索が完了する。

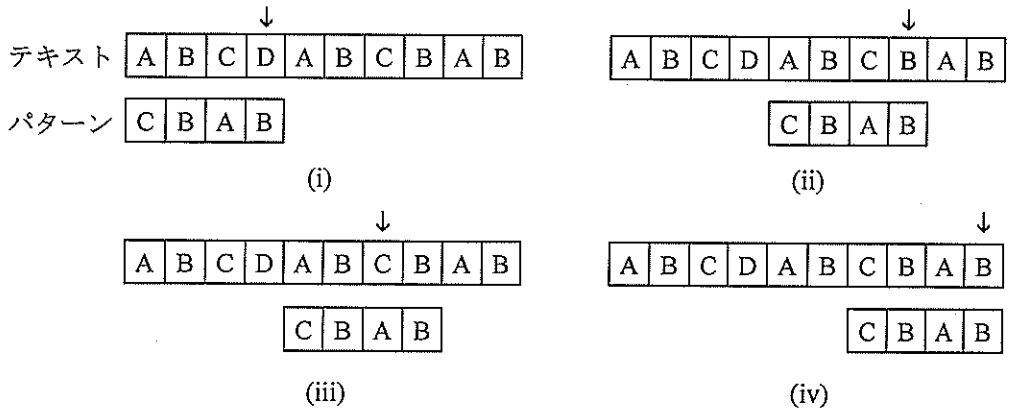

図 1

[次ページに続く]

プログラミング(2)　[3/3]

Program2

```
 1  #include <stdio.h>
 2  #include <string.h>
 3
 4  #define max(a, b) ((a) > (b) ? (a) : (b))
 5
 6  int bm_search(char *text, int t_len, char *pattern, int p_len) {
 7    int shift[256];
 8    int i, j;
 9
10    for(i = 0; i < 256; i++)
11      shift[i] = p_len;
12    for(i = 0; i < p_len - 1; i++)
13      shift[(unsigned char)pattern[i]] = p_len - i - 1;
14
15    i = [ ① ];
16    while(i < [ ② ]) {
17      printf("i=%d¥n", i);
18      j = [ ① ];
19
20      while(text[i] == pattern[j]) {
21        if(j == [ ③ ])
22          return i; /* Found. */
23        i--; j--;
24      }
25      i += max(shift[(unsigned char)text[i]], p_len - j);
26    }
27    return -1; /* Not Found. */
28  }
29
30  int main(void) {
31    char *text = "ABBCBCDABCDC";
32    char *pattern = "ABCD";
33    printf("Position=%d¥n",
34      bm_search(text, strlen(text), pattern, strlen(pattern)));
35    return 0;
36  }
```

(a) Program2 の空欄 ① ～ ③ を埋めて、プログラムを完成せよ。

(b) Program2 を実行したときの、標準出力への出力結果を示せ。

(c) BM 法では、例えば、テキストやパターンが “X” と “Y” の 2 種類の文字しか含まない場合のように、テキストやパターンを構成する文字の種類が少ないと、一般に検索の効率(efficiency)が低下する。その理由を簡潔に説明せよ。

(d) プログラム 25 行目では、テキストの比較位置を移動しているが、この行を単に

```
i += shift[(unsigned char)text[i]];
```

と記述すると、正しく文字列検索を行うことができない場合がある。その理由を簡潔に説明せよ。